Brainstormers 2D — Team Description 2009

T. Gabel and M. Riedmiller

Neuroinformatics Group
Institute of Mathematics and Computer Science
Institute of Cognitive Science
Universität Osnabrück, 49069 Osnabrück, Germany
{thomas.gabel|martin.riedmiller}@uos.de

Abstract. The main focus of the Brainstormers' effort in the RoboCup soccer simulation 2D domain is to develop and to apply machine learning techniques in complex domains. In particular, we are interested in applying reinforcement learning methods, where the training signal is only given in terms of success or failure. Our final goal is a learning system, where we only plug in "win the match" – and our agents learn to generate the appropriate behavior. Unfortunately, even from very optimistic complexity estimations it becomes obvious, that in the soccer simulation domain, both conventional solution methods and also advanced today's reinforcement learning techniques come to their limit – there are more than $(108 \times 50)^{23}$ different states and more than $(1000)^{300}$ different policies per agent per half time. This paper outlines the architecture of the Brainstormers team, focuses on the use of reinforcement learning to learn various elements of our agents' behavior, and highlights other advanced artificial intelligence methods we are employing.

1 Introduction

The Brainstormers project was established in 1998, starting off with a 2D team. Ever since we have been participating in RoboCup's annual soccer simulation tournaments. Over the years, the Brainstormers Tribots (competing in RoboCup's MidSize league since 2002), the Brainstormers 3D (soccer 3D simulation, 2004–2006), as well as the Brainstormers Twobots (Humanoid League, 2008) expanded the Brainstormers team. Our work has been accompanied by the achievement of several successes such as multiple World Vice Champion titles and the World Champion titles at RoboCup 2005 in Osaka (2D), RoboCup 2006 in Bremen (MidSize), RoboCup 2007 in Atlanta (2D + MidSize), and RoboCup 2008 in Suzhou (2D).

The team description paper at hand focuses on the Brainstormers 2D, our team competing in soccer simulation's 2D league. The underlying and encouraging research goal of the Brainstormers has always been to exploit AI and machine learning techniques wherever possible. Particularly, the successful employment of reinforcement learning (RL) methods for diverse elements of the Brainstormers' decision making modules – and their integration into the competition team – has been and is our main focus as shall be detailed in the subsequent sections.

1.1 Design Principles

The Brainstormers 2D rely on the following basic principles:

- There are two main modules: the world module and the decision making module.
- Input to the decision module is the approximate, complete world state as provided by the soccer simulation environment.
- The soccer environment is modelled as a Markovian Decision Process (MDP).
- Decision making is organized in complex and less complex behaviors.
- A large part of the behaviors is learned by reinforcement learning methods.
- Modern AI methods are applied wherever possible and useful (e.g. particle filters are used for improved self localization).

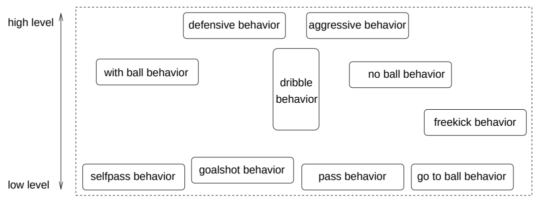

Fig. 1. The Behavior Architecture

1.2 The Brainstormers Agent

The decision making process the Brainstormers Agent is based upon is inspired by behavior-based robot architectures. A set of more or less complex behaviors realize the agents' decision making as sketched in Figure 1. To a certain degree this architecture can be characterized as hierarchical, differing from more complex behaviors, such as "no ball behavior", to very basic, skill-like ones, e.g. "pass behavior". Nevertheless, there is no strict hierarchical sub-divisioning. Consequently, it is also possible for a low-level behavior to call a more abstract one. For instance, the behavior responsible for intercepting the ball may, under certain circumstances, decide that it is better to not intercept the ball, but to focus on more defensive tasks and, in so doing, call the "defensive behavior" delegating responsibility for action choice to it. Our team's source code has been made publicly available and can be retrieved from our team web site[1].

2 Recent Research Efforts and Developments

After having outlined the basics on the Brainstormers' competition team, we now want to give an overview on recent developments and on one specific reinforcement learning approach that significantly enhanced our team's capabilities. A more comprehensive review of our efforts on utilizing neural reinforcement learning methods in the scope of the Brainstormers 2D can be found in [1].

[1] http://www.brainstormers.uos.de

2.1 Learning an Aggressive Defense Behavior

This section summarizes a defense scenario of crucial importance we addressed recently: The task of the agent is to disturb the opponent ball leading player, in particular, to avoid that it pulls ahead, and, if possible, to steal away the ball from him. A general strategy to achieve these goals is difficult to implement which is why we decided to employ a neural reinforcement learning approach that allows our players to train the hassling of opponent ball leading players. This approach is described in detail in [2]. Here, we provide a short summary of our *NeuroHassle* case study only, with focus on recent enhancements. It is worth noting that the utilization of the resulting aggressive defense behavior significantly improved our team's strength and substantially contributed to winning the World Champion titles of RoboCup 2007 and 2008.

2.1.1 Problem Formalization

The state space is 9-dimensional and covers, in a compressed form, information about positions and velocities of both players involved as well as of the ball. Additionally, some information is incorporated to indicate where on the playing field the current situation is located. The learning agent is allowed to use $dash(x)$ and $turn(y)$ commands where the domains of both commands' parameters ($x \in [-100, 100]$, $y \in [-180°, 180°]$) are discretized such that in total 76 actions are available to the agent at each time step.

Within the reinforcement learning framework we model the learning task as a terminal state problem with both terminal goal S^+ and failure states S^-. Intermediate steps are punished by constant costs of $c = 0.05$, whereas $J(s) = 0.0$ for $s \in S^+$ and $J(s) = 1.0$ for $s \in S^-$ by definition.

We have to distinguish between different types of goal and failure states.

Successes A duelling episode can be considered successful, i.e. finished by reaching a terminal state $s \in S_1^+$, if the ball has been brought into the learning player's kickable area or if it has managed to position in such a manner that issuing a tackle command yields a successful tackle for the ball with very high probability.

It may also happen that the ball leading opponent player simply kicks the ball away (usually forwards), as soon as the learning agent has approached or hassled him too much, or if it simply considers his situation to be too hopeless to continue dribbling. Consequently, if an opponent issues such a kind of a panic kick, the episode under consideration may be regarded as a semi-success, since the learning agent has managed to effectively interfere with the dribbler, though it has not conquered the ball (goal state set S_2^+).

Failures The ADB player is said to fail (entering a failure state $s \in S^-$), if the ball leading player has kept ball possession, has overrun the learning agent and escaped at least 7m from him, or approached the goal such that a goal shot might become promising.

We also distinguish episodes without a clear winner that were ended by a time-out (maximal episode duration of 35 time steps). We refer to such duels as semi-failures because the learning agent was not effective in interfering with the dribbling player – in a real match the ball leader may have had the chance to play a pass to one of its teammates within that time.

2.1.2 The Learning Algorithm

The learning agent starts with a cost-to-go function J_0 represented by a randomly initialized multilayer perceptron neural network. During interacting with the environment this function is always exploited greedily, i.e. realizing policy π_{k+1}, and simulated experience is collected. New estimates for $\hat{J}^{\pi}_{k+1}(s)$ are calculated according to

$$J_k^{target}(s) := E\{\sum_t c(s_t, \pi_k(s_t), s_{t+1}) | s_0 = s\}. \tag{1}$$

for successful episodes, failure states $s \in S^- \cap I$ in the experience set are associated with maximal costs of 1.0, and semi-success as well as semi-failure episodes (which play a negligible role as learning moves on) are disregarded for evaluating π_{k+1}.

Central to the learning process is that we perform neural network training in batch-mode: After having simulated a larger number of training episodes and, in so doing, having built up a set of representative states $I \subset S$, where for each $s \in I$ we have an estimated value $\hat{J}^{\pi}_{k+1}(s)$, the next cost-to-go function is determined by invoking the underlying batch supervised learning process. For neural network training we employ the back-propagation variant Rprop [3] using default parameters.

2.1.3 Discussion

In [2], we have shown and analyzed the learning curves for an exemplary learning experiment: The neural network representing the value function from which a greedy policy can be induced has been trained against the binary of WrightEagle for training situations located in the centre of the playing field. An initially clueless learning agent quickly improves its behavior and finally succeeds in successfully conquering the ball in more than 80% of all attempts.

Figure 2 visualizes the cost-to-go function acquired after 30000 training episodes for a small, two-dimensional fraction of the 9-dimensional state space: While zero object velocities and constant player body angles are assumed, the plot shows how desirable which position on the pitch would be for the learning agent. Obviously, positions where the learning player blocks the dribbler's way towards the goal are of high value, whereas high costs are to be expected, if the opponent has already overrun our player. From the shape of $\tilde{J}$ and the resulting equi-cost lines it can be concluded, that – from the learning agent's perspective – the most desirable direction into which to move is the one indicated by the gray-colored arrow. Accordingly, this kind of greedily exploiting $\tilde{J}$ by following the steepest ascent brings the learner onto a promising interception course (assuming a rational, i.e. ahead-moving opponent).

During the run of a standard game (6000 time steps), the players of our team start on average 66 duelling episodes. Therefore, even under the very conservative

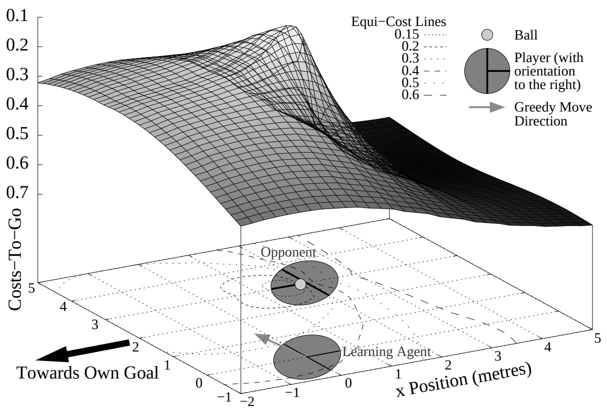

Fig. 2. Learned value function for the duelling behavior.

assumption that only about half of all attempts are successful, we can draw the conclusion that the learned behavior allows for conquering the ball at least 30 times per game. As pointed out, employing the learned *NeuroHassle* policy was one of our crucial moves for winning the World Championships tournament RoboCup 2007/2008 in Atlanta/Suzhou.

2.2 Using RSPSA to Overcome the Opponent's Offside Trap

When trying to conquer the opponent's offside line by coordinated pass playing and player movements, it is of crucial importance that the player who is about to receive the OOOT pass – the pass which is played beyond the opponent team's offside line and, hence, represents a pass that is meant to overcome the opponent's offside trap – starts running into the correct direction at the right point of time. Preferably, this player should be positioned right before the offside line while running at its maximal velocity at the moment when the pass is being played, i.e. when the ball leaves the pass playing teammate's kickable area.

For this to happen, the pass receiving player must follow a clever k-step movement policy bringing it towards the offside line. Here k is meant to have a small value (typically, not larger than 5 steps) and this short-time policy is being started when the ball leading teammate announces that it will soon play an OOOT pass. We developed an approach to learn such k-step movement policies online using the stochastic approximation method RSPSA, which is a combination of the well-known simultaneous perturbation stochastic approximation (SPSA [4]) procedure with the resilient propagation (Rprop [3]) learning technique. The results obtained so far were encouraging as well as interesting from the perspective of using stochastic optimization approaches for real-time policy learning. So far, however, the learning routines were not yet incorporated into the competition team. More details on this approach can be found in [5].

2.3 Flash Animations of RoboCup 2D Soccer Simulations

Watching logfiles of recorded 2D soccer simulation matches using the standard `rcssmonitor` and `logplayer`, sometimes leaves the unacquainted viewer puzzled. Hence, providing animations that, at least in part, resemble real soccer is a must. For these reasons, it was our goal to provide a better looking and more realistic animation of a simulated soccer game. The result of these effort is a tool called `rcg2swf` [6] that is capable of creating a handsome visualization of a RoboCup Soccer 2D Simulation game. A comprehensive of this tool including all its nice features can be found on our team's website[2]. Of course, `rcg2swf` is freely available and can be downloaded from that URL as well.

3 Summary

During the period from 2004 to 2007, our team could greatly benefit from facing a nearly stable simulation environment. This allowed us to concurrently (a) redesign vast parts of our team play and (b) enhance several of the machine learning approaches we employ in such a manner that the resulting behaviors are highly competitive. In 2008 as well as in this year, a considerable number of major changes is being introduced to the simulation environment (Soccer Server Ver.12/13). As a consequence, our main focus in 2009 is to adapt our team and coach to the changes introduced. Unfortunately, these changes and the required adaptations will bind most of our resources. Consequently, the development and realization of new ideas, concepts, or novel learning approaches will most likely be rendered impossible.

References

1. Riedmiller, M., Gabel, T.: On Experiences in a Complex and Competitive Gaming Domain: Reinforcement Learning Meets RoboCup. In: Proceedings of the 3rd IEEE Symposium on Computational Intelligence and Games (CIG 2007), Honolulu, USA, IEEE Press (2007) 68–75
2. Gabel, T., Riedmiller, M., Trost, F.: A Case Study on Improving Defense Behavior in Soccer Simulation 2D: The NeuroHassle Approach. In: RoboCup 2008: Robot Soccer World Cup XII, LNCS, Berlin, Springer Verlag (2008, to appear)
3. Riedmiller, M., Braun, H.: A Direct Adaptive Method for Faster Backpropagation Learning: The RPROP Algorithm. In Ruspini, H., ed.: Proceedings of the IEEE International Conference on Neural Networks (ICNN), San Francisco (1993) 586–591
4. Spall, J.: Adaptive Stochastic Approximation by the Simultaneous Perturbation Method. IEEE Transactions on Automatic Control **45** (2000) 1839–1853
5. Wolowik, E.: Hybride Ansätze zum Optimieren kooperativer Angriffsstrategien. Diploma thesis, University of Osnabrueck (2008)
6. Schwegmann, T.: Flashanimationen von RoboCup Soccer 2D Simulationen. Diploma thesis, University of Osnabrueck (2007)

[2] http://www.ni.uos.de/index.php?id=1024